(2) Cited Reference

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 07-191872
(43)Date of publication of application : 28.07.1995

(51)Int.Cl. G06F 11/28
G06F 9/46
G06F 11/32

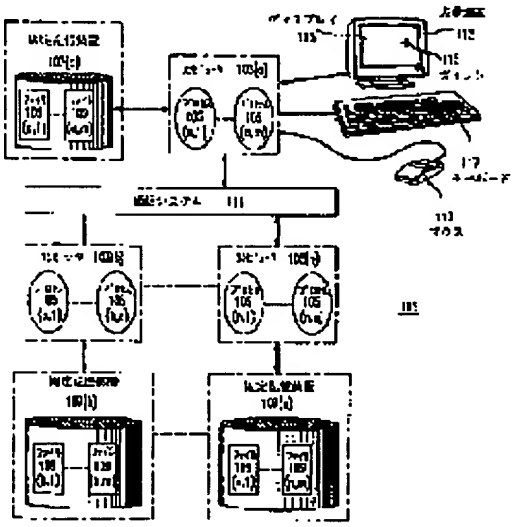
(21)Application number : 06-308467
(22)Date of filing : 18.11.1994

(71)Applicant : AT & T CORP
(72)Inventor : FOWLER GLENN S
KORN DAVID G
KOUTSOFIOS ELEFTERIOS
NORTH STEPHEN C

(30)Priority
Priority number : 93 154771 Priority date : 19.11.1993 Priority country : US

(54) SYSTEM ANALYZING DEVICE

(57)Abstract:
PURPOSE: To provide a method for analyzing and controlling the operation of the system of a cooperative process.
CONSTITUTION: A system calling library used by this program executes system calling, and this is displaced with a dynamic link library which transmits a message indicating that the system calling is executed. The message is received by a display system 112. The display system 112 generates picture display indicating the present state of the system of a process in order to respond to this message. The picture display displays the system of the process like a tree. A node in the tree indicates the process in the system, and a resource such as a file used by this process. An edge in the tree indicates relation between the process and another process or the resource. A user can control which system calling generates the message, the speed of the display system required for responding to the message, and the execution of the process.



LEGAL STATUS

[Date of request for examination] 25.04.1997

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]
[Patent number] 2883826
[Date of registration] 05.02.1999
[Number of appeal against examiner's decision of rejection]
[Date of requesting appeal against examiner's decision of rejection]
[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平7－191872

(43)公開日　平成７年(1995)７月28日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　11/28		Ａ　9290－5Ｂ		
9/46	３３０	Ｃ　7629－5Ｂ		
11/32		Ａ　9290－5Ｂ		

審査請求　未請求　請求項の数27　ＦＤ　（全 13 頁）

(21)出願番号　特願平6－308467

(22)出願日　平成６年(1994)11月18日

(31)優先権主張番号　１５４７７１

(32)優先日　1993年11月19日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　390035493
エイ・ティ・アンド・ティ・コーポレーション
ＡＴ＆Ｔ　ＣＯＲＰ.
アメリカ合衆国　10013－2412　ニューヨーク　ニューヨーク　アヴェニュー　オブ　ジ　アメリカズ　32

(72)発明者　グレン　ステファン　ファウラー
アメリカ合衆国、07076　ニュージャージー、スコッチ　プレインズ、ライド　プレイス　2322

(74)代理人　弁理士　三俣　弘文

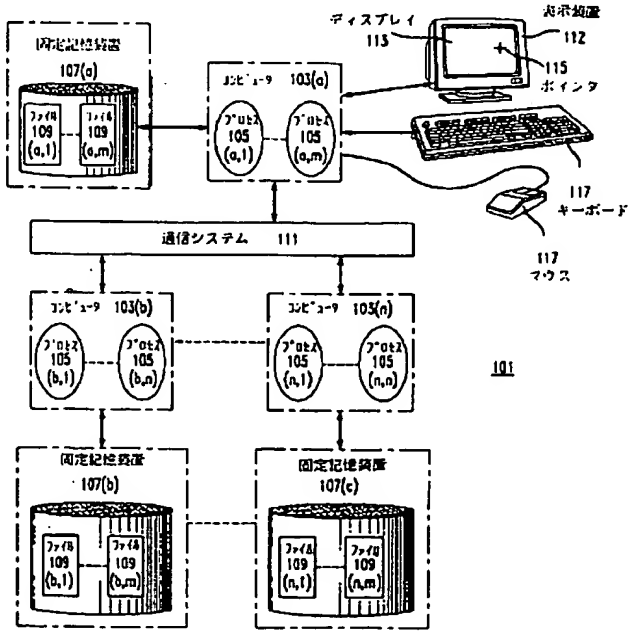
最終頁に続く

(54)【発明の名称】　システム解析装置

(57)【要約】

【目的】　共同プロセスのシステムの動作を解析・制御する方法を提供する。

【構成】　このプログラムにより使用されるシステム呼出のライブラリは、システム呼出を実行し、システム呼出が実行されたことを示すメッセージを送信する動的リンクライブラリにより置換される。メッセージは表示システムにより受信される。表示システムは、プロセスのシステムの現行状態を示す画像表示を生成することによりこのメッセージに応答する。画像表示はプロセスのシステムをツリー状に表示する。ツリー内のノードはシステム内のプロセスと、このプロセスにより使用されるファイルのような資源を示す。ツリー内のエッジはプロセスと他のプロセス又は資源との間の関係を示す。ユーザはどのシステム呼出がメッセージを生成するか制御でき、表示システムがメッセージに応答する速度を制御でき、プロセスの実行を制御できる。



1

【特許請求の範囲】

【請求項１】　プロセスにより実行されるオペレーティング・システム呼出した結果としてのメッセージを、少なくとも一つのプロセスに送信させるモニタ手段と，プロセスのシステムの状態を示す画像表示を生成する手段と，からなることを特徴とするプロセスのシステム解析装置。

【請求項２】　前記画像表示は、プロセスのシステムを有向非巡回グラフとして表示し、プロセス及びプロセスにより使用される資源はグラフ内のノードである請求項１の装置。

【請求項３】　モニタ手段は、オペレーティング・システム呼出を行い、かつ、メッセージを送信するルーチンを更に含む請求項１の装置。

【請求項４】　ルーチンは、プロセスにより実行されるプログラムと動的にリンクされる請求項３の装置。

【請求項５】　モニタ手段は、ルーチンがメッセージをどのように処理するか決定するマスク手段を更に含み、オペレーティング・システム呼出を行うルーチンは、マスク手段により指示されるように、メッセージを更に処理する請求項３の装置。

【請求項６】　マスク手段は、メッセージを送信すべきか否か示す第１のマスク手段を有する請求項５の装置。

【請求項７】　マスク手段は、メッセージに関するフォームを示す第２のマスク手段を含む請求項５の装置。

【請求項８】　モニタ手段は肯定応答メッセージを受信することができ，マスク手段は、送信されたメッセージを肯定応答すべきか否か示す第３のマスク手段を有し，オペレーティング・システム呼出を行うルーチンは、送信されたメッセージが肯定応答されるべき場合、送信されたメッセージは肯定応答されるべきであり、該肯定応答メッセージを待機中であることを示すことにより第３のマスク手段に応答する請求項５の装置。

【請求項９】　各プロセスに対応するマスク手段が存在する請求項５の装置。

【請求項１０】　子プロセスはその親プロセスに対応するマスク手段のコピーを受信する請求項９の装置。

【請求項１１】　モニタ手段はマスクメッセージを受信することができ，かつ、送信されたメッセージをルーチンがどのように処理するか、マスクメッセージに応じて、マスク手段に決定させることによりマスクメッセージに応答することができる請求項９の装置。

【請求項１２】　メッセージに応答する手段がメッセージに応答する方法を制御する手段を更に含む請求項１の装置。

【請求項１３】　モニタ手段からメッセージを受信する手段を更に有し、

メッセージに応答する手段はメッセージ受信手段からのメッセージを読出し、

メッセージ応答手段がメッセージに応答する方法を制御

2

する手段は、メッセージ応答手段がメッセージを読出す方法を制御することにより制御を行う請求項１２の装置。

【請求項１４】　装置のユーザに応答して、制御メッセージを送信する手段を更に有し、

モニタ手段は、制御メッセージにより必要とされるような挙動をプロセスにとらせることにより、制御メッセージに更に応答する請求項１の装置。

【請求項１５】　モニタ手段は、オペレーティング・システム呼出を行い、メッセージを送信するルーチンを更に含み、

ルーチンは実行を終了するために制御メッセージを待機している請求項１４の装置。

【請求項１６】　モニタ手段は、送信されたメッセージを肯定応答すべきか否か示すマスク手段を有し、

オペレーティング・システム呼出を行うルーチンは、送信されたメッセージが肯定応答すべきものである場合に、該送信メッセージは肯定応答すべきものであることを該メッセージ内に示すことによりマスク手段に応答し、

送信メッセージに応答する手段は、送信メッセージが肯定応答すべきものである場合に、その指示をユーザに供給することにより送信メッセージに応答し、

ユーザはこの指示に応答して制御メッセージを送信する請求項１５の装置。

【請求項１７】　子プロセスを生成する場合又は資源にアクセスする場合、少なくとも一つのプロセスにメッセージを送信させる手段、及びこのメッセージに応答して、画像表示を発生させる手段からなり、

前記画像表示は有向非巡回グラフとしてプロセスのシステムを表示し、

プロセス及びプロセスにより使用される資源はグラフ内のノードであることを特徴とするプロセスのシステム解析装置。

【請求項１８】　グラフ内の第１のエッジは或る一つのプロセスと或る一つの資源との間の関係を示す請求項２又は１７の装置。

【請求項１９】　グラフ内の第２のエッジは、最初の或る一つのプロセスと、この最初の或る一つのプロセスの子孫である次の或る一つのプロセスとの間の関係を示す請求項２又は１７の装置。

【請求項２０】　システム内のプロセスは１個以上のプロセッサで実行し、

所定の或る一つのプロセッサで実行するプロセス用のノードは所定のプロセッサを示すクラスタに属する請求項２又は１７の装置。

【請求項２１】　ノードの形状は、ノードが或る一つのプロセス又は或る一つの資源を示すか否か指摘する請求項２又は１７の装置。

【請求項２２】　資源はファイル資源と通信資源を含

み、
ファイル資源用のノードは第１の形状を有し、通信資源用のノードは第１の形状と異なる第２の形状を有する請求項２１の装置。
【請求項２３】　メッセージは一連のメッセージのうちの一つであり、
画像表示発生手段は、メッセージにより必要とされるように画像表示を変更する請求項２又は１７の装置。
【請求項２４】　コンピュータ内で実現され、かつ、第１のルーチンの第１のライブラリを使用するシステムの動作を解析するための解析装置であり、
第１のルーチンと同じ機能を果たし、更に、システムの変更を指示するメッセージを生成する第２のルーチンの第２のライブラリ，第１のライブラリ用の実行可能ノードを第２のライブラリ用の実行可能ノードと動的に置換する手段，第２のルーチンがメッセージをどのように生成するか指示する、第２のルーチンにアクセス可能なマスク手段、第２のルーチンは、該ルーチン内に指示されたようにメッセージを生成することによりマスク手段に応答する，メッセージに応答して、システムの状態を出力する手段，からなることを特徴とする解析装置。
【請求項２５】　システムの状態はシステムの画像表示として出力される請求項２４の解析装置。
【請求項２６】　システムの画像表示は、有向非巡回グラフとしてシステムを表示し、システム内のエンティティは、グラフにノードとして現れ、エンティティ間の関係はグラフにエッジとして現れる請求項２５の解析装置。
【請求項２７】　解析装置のユーザが該装置用の制御メッセージを供給する手段を更に有し、
第２のライブラリはルーチンを受信する制御メッセージを含み、
第２のルーチンは制御メッセージに応答する請求項２４の解析装置。
【発明の詳細な説明】
【０００１】
【産業上の利用分野】本発明はシステムの解析及び制御に関する。更に詳細には、本発明はコンピュータで実行するプロセスを構成するシステムの解析及び制御に関する。
【０００２】
【従来の技術】巨大なシステムを構築する際の重要な事項はデバッグである。すなわち、システムの実現においてエラーを検出し、解析し、そして訂正することである。システムがコンピュータで実行されるプログラムにより実現される場合、プログラムをデバッグするのに利用可能な様々なツールが存在する。ｃｉａのようなディスカバリプログラムが存在する。
【０００３】このプログラムは、プログラマが、プログラムがどのように構成されているか理解するのを助ける。このプログラムはデバッガと呼ばれる。このデバッガは、デバッグされているプログラムが実行されるときに起こる事象の内容をプログラマに見せることができる。最近のデバッガは、プログラマが、デバッグされているプログラムの実行を対話型で制御できるようになっている。

【０００４】このようなデバッガの一例は、フリーソフトウエア財団から入手できる、ＧＤＢデバッガである。プログラムの呼出履歴，分散メモリの並列プログラムにより発生されたイベント又はプログラムの並列実行のトレースなどのような情報を示すために、デバッガは更にグラフィック・インターフェースの使用を開始した。
【０００５】このようなグラフィック・インターフェースの一例は、ＩＥＥＥコンピュータ，１９９３年６月発行，８８－９５頁に掲載された、アダム・ベグエリン(AdamBeguelin)らの、“複合環境におけるビジュアリゼーション及びデバッギング(Visualization and Debugging in a Heterogeneous Environment)”と題する論文中に開示されている。
【０００６】ここで述べられているディスカバリプログラム及びデバッガはその本来の用途には完全に適合する。しかし、最近のシステムは一般的に、一連の共同サブルーチンとして実現されるのではなく、むしろ、一連の共同プロセスとして実現される。ここで、説明の便宜上、“プロセス”とは、ユーザ用のプログラムを実際に実行するコンピュータシステムにおける実体（エンティティ）として定義する。
【０００７】多くのシステムでは、共同プロセスは異なるコンピュータで実行する。システムが一連の共同プロセスとして実現される場合、システムのデバッギングは、このプロセスにより実行されている個々のプログラムの理解とデバッギングだけでなく、プロセスの共同動作の理解とデバッギングも含む。このプロセスの共同動作の理解とデバッギングは、ここに述べたプログラムディスカバリツール及びデバッガによっては行うことができない。
【０００８】現今のコンピュータシステムは、共同プロセスを構成するデバッギングシステムのために貧弱な資源しか提供しない。例えば、ＵＮＩＸオペレーティング・システム（ＵＮＩＸはユニックス・システム・ラボラトリース社の登録商標である。）を使用するコンピュータシステムでは、オペレーティング・システムに対してプロセスにより為された呼出のリストを出力する“トレース”ユーティリティが存在する。
【０００９】また、どんなファイルが所定のプロセスをオープンするかユーザに伝える“オーファイル(ofile)”ユーティリティ及びどんなプロセスが所定のファイルを使用しているか識別する“エフユーザ(fuser)”ユーティリティも存在する。これらの貧弱なデバッギングツールの欠点は、単一のプロセッサで実行するプロセス

に関する情報しか提供しないことであり、従って、共同プロセスを異なるプロセッサで実行する理解及びデバッギングシステムでは有用性が殆ど無い。

【００１０】

【発明が解決しようとする課題】従って、本発明の目的は前記のようなシステムの精密な解析と制御を可能にする方法を提供することにより、共同プロセスを構成するデバッギングシステムに伴う前記の問題点を解決することである。

【００１１】

【課題を解決するための手段】本発明の方法は、ビジュアルプロセスマネージャで体現される。ビジュアルプロセスマネージャはモニタとグラフィカル・ディスプレイ・ジェネレータ（画像表示発生器）を有する。モニタは、システム内のプロセスが子プロセスを作成するか、又はファイルのような資源にアクセスするような操作に関するオペレーティング・システム呼出を実行する場合に、メッセージを送信する。

【００１２】グラフィカル・ディスプレイ・ジェネレータはこのメッセージに応答して表示を生成する。この表示は、プロセスのシステムの状態及びメッセージにより必要とされるような変更を示す。従って、ユーザはプロセスのシステム内で起こっている事象を、この表示から決定できる。更に、ユーザは、この表示及びプロセスのシステムさえも制御するために、マウスのようなポインティング・デバイスを使用することもできる。

【００１３】好ましい実施例では、この表示はノード及びエッジの有向非巡回グラフである。アクセスするプロセス及び資源はグラフ内のノードであり、ノードを接続するエッジはノード間の関係を示す。例えば、プロセスを示すノードはエッジによりその親用のノードに接続され、プロセスによりアクセスされるファイルに関するノードは別のエッジによりプロセス用のノードに接続される。

【００１４】ビジュアルプロセスマネージャの重要な利点は、プロセスにより実行されるコードの変更が不要なことである。モニタは動的リンク・ライブラリにより実現される。動的リンク・ライブラリは、コンピュータシステムにより提供されたオペレーティング・システム呼出のライブラリを、システム呼出を行うだけでなく、メッセージも生成する新たなライブラリに置換する。

【００１５】モニタ内のプロセス毎のマスクにより、ユーザは、どのシステム呼出がメッセージを生じるか、メッセージのフォームはどのようなものであるべきか、及びシステム呼出用のライブラリルーチンは継続前にユーザからの肯定応答を待つべきか否かを指示することができる。

【００１６】好ましい実施例は、プロセスのシステムを解析するために本発明の方法を使用するが、本発明の方法は、一般的に適用可能であり、ルーチンのライブラリ

が、同じ機能を果たすが副作用としてメッセージも送信する動的リンク・ライブラリにより置換され得るような全ての状況において使用することができる。本発明の特別な利点は、システムを、システムのアプリケーションレベルコードの変更又はリコンパイルすること無しに解析できることである。

【００１７】

【実施例】以下、図面を参照しながら本発明を具体的に説明する。

【００１８】先ず、本発明を実現するシステムについて説明し、次いで、システムの好ましい実施例について説明し、最後に好ましい実施例の内容を具体的に説明する。

【００１９】ビジュアルプロセスマネージャを有するシステム：図１

図１は常用の最新型分散コンピュータシステムの模式的構成図である。数台のプロセッサ１０３は通信システム１１１を介して相互に通信する。通信システム１１１は、メッセージをプロセッサ１０３間で送信できるものであればどのようなシステムであってもかまわない。各プロセッサの実行プログラムは多数のプロセス１０５である。

【００２０】プロセス１０５はメッセージを、同一の又は他のプロセッサ１０３で実行する他のプロセスに送信できる。プロセス１０５は固定記憶装置１０７内のファイル１０９ともアクセスできる。ファイルは、プロセス１０５を実行するプロセッサ１０３に属する固定記憶装置１０７に属するか又は別のプロセッサ１０３に属する固定記憶装置１０７に属することができる。

【００２１】ビジュアルプロセスマネージャのユーザは、表示装置１１２と、何れかのプロセッサ１０３（この場合は、プロセッサ１０３（ａ））に接続されたキーボード１１７及びマウス１１９のような入力装置を有する。キーボード１１７又はマウス１１９により、ユーザは表示装置１１２のディスプレイ上の表示に応答してプロセッサ１０３（ａ）に入力を送ることができる。更に詳細には、ユーザはマウス１１９を用いてディスプレイ１１３内のポインタ１１５を移動させることができ、また、マウス１１９のボタンを使用してコマンド入力を行うことができる。

【００２２】ビジュアルプロセスマネージャのユーザビュー：図２，図３及び図５

ビジュアルプロセスマネージャのユーザは一般的に、図２に示されるようなディスプレイにより作業する。グラフ２０１は表示装置１１２上に現れる。大抵の場合、プロセッサ１０３はウインドウシステムを実行し、グラフ２０１はディスプレイ１１３内のウインドウ内に現れる。グラフ２０１は有向非巡回グラフである。

【００２３】このグラフは４台のコンピュータで実行する共同プロセスのシステムを示す。各コンピュータは、

コンピュータの名称（例えば、キーウィ(kiwi)）を有するクラスタ２０３で示される。各クラスタの内部は一群のプロセスボックス２０５である。各プロセスボックス２０５は、クラスタにより示されるコンピュータで実行するプロセス１０５を示す。プロセスボックス内の番号は、このプロセスに関するプロセス識別子（ｐｉｄ）である。

【００２４】プロセス識別子は、プロセスを識別するためにコンピュータ２０３のオペレーティング・システムにより使用される番号である。文字列はプロセスにより現に実行されているプログラムの名称である。従って、ｐｉｄ２３０８７を有するプロセス１０５はｇｃｃプログラムを実行する。？を有するプロセスボックス２０５のプロセス１０５はプロセス初期化プログラムを実行する。点線２０７は、プロセス１０５間の親子関係を示す。

【００２５】親プロセスが他のプロセス１０５を生成するオペレーティング・システムコマンドを実行する場合、プロセス１０５は別のプロセス１０５の親である。従って、プロセス２３０８６はプロセス２３０８８の親である。

【００２６】言うまでもなく、プロセス１０５はファイル１０９にアクセスできる。ファイルはファイル長円２０９としてグラフ２０１に現れる。各ファイル長円２０９は、ファイル長円２０９により示されるファイルの名称である文字列を有する。プロセス１０５がファイル１０９をオープンした場合、このプロセスのプロセスボックス２０５とこのファイルのファイル長円２０９を接続する実線のエッジ２１３が存在する。

【００２７】プロセス１０５が読出用のファイルをオープンした場合、矢線の先頭はプロセスボックス２０５に存在する。プロセス１０５が書込用のファイルをオープンした場合、矢線の先頭はファイル長円２０９に存在する。読出用及び書込用の両方のファイルがオープンされた場合、矢線の先頭はエッジ２１３の両端に存在する。

【００２８】プロセス１０５は、本発明が使用されるシステム内で、ＵＮＩＸオペレーティング・システムにより提供されるパイプ及びソケットにより通信する。パイプは、同じコンピュータ１０３で実行するプロセス間の通信にのみ使用され、ソケットは、異なるコンピュータ１０３で実行するプロセス間の通信に使用される。ソケットの場合、通信は実際に通信システム１１１を介して伝わる。パイプ及びソケットは、パイプ環２１５及びソケット環２１１としてグラフ２０１に現れる。

【００２９】ソケット環２１１は、ソケットが生成された時のソケットの受け手であったコンピュータの名称及び受け手コンピュータにおけるポートの番号を有する。ファイル長円２０９の場合と同じ方法で、エッジ２１３はパイプ環２１５及びソケット環２１１と一緒に使用される。すなわち、パイプ又はソケットが書込のためにオープンされる場合、矢線の先頭は環２１１又は２１５に存在し，パイプ又はソケットが読出のためにオープンされる場合、矢線の先頭はプロセスボックス２０５に存在する。

【００３０】好ましい実施例が実現されるＵＮＩＸオペレーティング・システムのバージョンでは、パイプは片方向であり、或るプロセスはパイプに書込み、他のプロセスはパイプから読出す。これに対して、ソケットは双方向である。

【００３１】ビジュアルプロセスマネージャの動作：図３及び図５

好ましい実施例では、ビジュアルプロセスマネージャは２つの主要構成要素からなる。一つは、プロセスの実行に応じて、システム内のプロセスをモニタする構成要素であり、もう一つは、モニタ用構成要素からメッセージを受信し、このメッセージに応じてグラフ２０１を生成する構成要素である。

【００３２】好ましい実施例における構成要素は、１９９３年に米国のシンシナティで開催されたＵＳＥＮＩＸ夏季会議講演集の２７９～２９０頁に掲載された、グレン・フォウラー(Glenn Fowler)らの“ユーザレベル複製ファイルシステム(A user-level replicated file system)”と題する論文に記載された、ｎ次元ファイルシステム（ｎＤＦＳ）に基づくモニタと、プログラムｄｏｔ（ＩＥＥＥトランザクションズ・オン・ソフトウエア・エンジニアリング、Ｖｏｌ．１９，Ｎｏ．３（１９９３年３月）に掲載されたエムデン(Emden)らの“有向グラフの作図方法(ATechnique for Drawing Directed Graphs)”と題する論文に説明されている）及びｌｅｆｔｙ（カナダ，アルバータ州のカルガリにおける９１年度グラフィックス・インターフェース(Graphics Interface)，６８～７６頁（１９９１年）に掲載された、エレフシェリオス・コウトソフィオス(Eleftherios Koutsofios)らの“Ｌｅｆｔｙ：テクニカルピクチャ用の２ビューエディタ(a two-view editor for technical pictures)”と題する論文中に開示されている）に基づくディスプレイジェネレータである。

【００３３】好ましい実施例において、ビジュアルプロセスマネージャの動作を開始するために、ユーザはディスプレイジェネレータを実行する第１のプロセスと、モニタを実行する第２のプロセスをスタートさせる。各プロセスは表示装置１１２内にそれ自身のウインドウを有する。第２のプロセスが実行されると、ユーザは、ログファイルを指定する、第２のプロセスのシェル内のセットアップコマンドを実行する。

【００３４】モニタはこのログファイルにメッセージを送り、また、ディスプレイジェネレータはこのログファイルからその入力と、モニタにより監視されるべきオペレーティング・システムのシステム呼出を獲得する。好ましい実施例では、ユーザは、プロセス関連オペレーテ

ィング・システム呼出と、プロセス関連オペレーティング・システム呼出及びファイルアクセス関連オペレーティング・システム呼出、及び前記の二つと共にＩ／Ｏシステム呼出だけを指定できる。

【００３５】好ましい実施例では、プロセス関連オペレーティング・システム呼出はＵＮＩＸフォーク（ｆｏｒｋ），実行（ｅｘｅｃ）及びエグジット（ｅｘｉｔ）呼出であり、ファイルアクセス関連呼出はオープン（ｏｐｅｎ），クローズ（ｃｌｏｓｅ），ダップ（ｄｕｐ）及びパイプ（ｐｉｐｅ）であり、Ｉ／Ｏ呼出は読出（ｒｅａｄ）と書込（ｗｒｉｔｅ）である。

【００３６】前記のセットアップコマンドが実行されると、ディスプレイジェネレータ及びセットアップコマンドを実行するプロセスの現行状態により、モニタはログファイル読出へのメッセージ送信を開始し、そしてプロセスの子孫がグラフ２０１に表示される。従って、プロセスの所定のシステムをモニタするのに必要なことは、セットアップコマンドが実行されるプロセスの子としてシステムの第１のプロセスをスタートさせることだけである。

【００３７】ディスプレイジェネレータはリアルタイムモードとシングルステップモードの２つのモードで動作する。リアルタイムモードでは、ディスプレイジェネレータは、モニタがメッセージをログファイル内に配置するのに応じて、ログファイル内のメッセージを読出す。シングルステップモードでは、モニタされるシステムは、モニタが終了するまで実行される。この場合、モニタからの全てのメッセージはログファイル内に配置される。

【００３８】その後、ディスプレイジェネレータはログファイル内のメッセージを読出す。好ましい実施例では、モードは、ディスプレイジェネレータの実行を開始するコマンドのオプションにより指定される。いずれの場合も、グラフ２０１は、メッセージがログファイルから読出されるのに応じて、動的に更新される。

【００３９】何れかのモードにおけるディスプレイジェネレータの動作は、ポインタ１１５がグラフ２０１のノード上に配置されず、マウスの右ボタンがクリックされた場合に、グラフ２０１に現われるグローバルメニューにより制御される。図３はリアルタイムモードに関するグローバルメニューを示す。メニューオプションは次の通りである。

【００４０】●ドゥ・レイアウト(do layout)：グラフ２０１のレイアウトをやり直す，

●リドロー(redraw)：グラフをリフレッシュする，

●セーブ・グラフ(save graph)：グラフをファイルにセーブする，

●ズームイン(zoom in)：グラフを拡大する，

●ズームアウト(zoom out)：グラフを縮小する，

●クリーンアップ(cleanup)：デッドプロセスを示すノードをグラフ２０１から削除する，

●ファインド・ノード(find node)：ディスプレイをスクロールし、ｐｉｄ又は名称により指定されるノードをグラフ２０１のディスプレイの中央に配置させる，

●スタート／ストップ(start/stop)：アイテムが再び選択されるまで、グラフの更新を停止する，

●テキストビュー(textview)：ディスプレイジェネレータにより現に解読されているプログラムを調べる，及び

●エグジット(exit)：ディスプレイジェネレータの実行終了。

【００４１】オプションはマウスを使用することにより選択される。すなわち、マウスでポインタ１１５をメニュー内の所望のエントリに移動させ、そしてマウスの左ボタンをクリックすることにより、オプションを選択できる。グラフ２０１が、グラフ２０１の表示されているウインドウよりも大きい場合、ユーザはウインドウ上のスクロールバーを使用し、グラフの非表示部分をウインドウ内に持ってくることができる。

【００４２】シングルステップモードのグローバルメニューは、スタート／ストップ及びクリーンアップが次の二つの選択肢により置換される点で、リアルタイムモードと異なる。

【００４３】●プレイ・ノンストップ(play non-stop)：ディスプレイジェネレータは停止することなくログファイル内のメッセージに応答する，

●プレイ・アンティル(paly until)：ユーザはシステム呼出名称又はメッセージ一連番号を指定する，ディスプレイジェネレータは指定されたシステム呼出名称又はメッセージ一連番号が到着するまでログファイル内のメッセージに応答する。

【００４４】プレイ・ノンストップ又はプレイ・アンティルのどちらも選択されない場合、ユーザはメッセージがマウスにより読出される速度を制御する。ユーザがマウスの左ボタンをクリックすると、次のメッセージが読出され、グラフ２０１にそのメッセージに従って変更される。従って、シングルステップモードは、ユーザに、ユーザの関心のあるプロセスのシステムの動作部分だけを見せるオプションを与える。

【００４５】グラフ２０１のノードに関するメニューにより、ユーザは、個々のノードに関する情報を発見することができ、また、プロセスボックス２０５の場合には、プロセスを操縦することができる。ノードメニューを得るために、ユーザはポインタ１１５を所望のノードに移動し、そしてマウスの右ボタンをクリックする。ノードメニューの内容はノードの種類及びディスプレイジェネレータがリアルタイムモード又はシングルステップモードで動作されているか否かに応じて変化する。

【００４６】図５は、リアルタイムモードにおけるプロセスボックス６６７８に関するノードメニュー５０１を示す。可能なメニュー選択は次の通りである。

【００４７】●プリント・エントリ(print entry)：ノードにより示される実体に関する情報を印刷する，プロセスノードでは、この情報はノード種類，プロセスが実行されるマシンの識別名(id)，マシンの名称，プロセスのｐｉｄ，プロセスにより現に実行されているプロセスのパス名などである，
●プリント・アトリビュート(print attribute)：色，形状，フォント名及びフォントサイズなどのノードに関するディスプレイ属性を印刷する，
●キル(kill)：プロセスに対するＵＮＩＸオペレーティング・システムのキルコマンドを実行し、これを停止させる，プロセスはこのコマンドを無視することもできる，
●キル－９(kill-9)：プロセスを停止させるのに必要な形でキルを実行する，
●ランｄｂｘ(run dbx)：プロセスにより実行されているプログラム上でデバッガを実行させる。
【００４８】シングルステップモードでは、プリント・エントリ(print entry)及びプリント・アトリビュート(print attr)の選択肢だけが利用できる。
【００４９】ファイル長円２０９に関するメニューは、両方のモードにおいて、プリント・エントリ(print entry)及びプリント・アトリビュート(print attr)の選択肢だけを有する。プリント・エントリされる場合、ファイル名，デバイス名及びノード名が表示される。パイプ及びソケット環２１１に関するメニューはこれらの選択肢の他に、ショー・メッセージ(show message)選択肢も含む。
【００５０】このショー・メッセージ(show message)選択肢は、モニタがＩ／Ｏシステム呼出を監視している時に、選択されたパイプ又はソケットを通過するメッセージを示す。パイプに関するプリント・エントリの場合、識別子が表示され、これにより、パイプを使用する各プロセスはそのパイプを知る。ソケットの場合、ソケットの名称，送信元マシーン名，受信元マシーン名及びポートが表示される。
【００５１】実施例の具体的内容：図４，図６及び図７
図４は、ビジュアルプロセスマネージャの好ましい実施例の具体的構成を示す概要ブロック図である。ビジュアルプロセスマネージャ４０１は前記のようなモニタ４１８とディスプレイジェネレータ４１９の２つの主要構成要素を有する。モニタ４１８は、検査されているシステムに属するプロセス１０５の挙動を監視する。
【００５２】ディスプレイジェネレータ４１９は、表示装置１１３上にディスプレイ２０１を生成し、キーボード１１７及びマウス１１９からの入力に応答する。構成要素間の通信はログファイル４１１を介して行われる。モニタ４１８は監視されているプロセス１０５から受信したメッセージをログファイル内に書込み、ディスプレイジェネレータ４１９はログファイルからメッセージを読出す。
【００５３】モニタ４１８
モニタ４１８について更に詳細に説明する。モニタ４１８の構成要素は、検査されているシステムに属する各プロセス１０５用のメッセージ・ジェネレータ４０３と後置型サーバプロセス４０９からなる。プロセス１０５が、監視されているシステム呼出を実行する場合、メッセージ・ジェネレータ４０３はメッセージを生成する。
【００５４】後置型サーバプロセス４０９はプロセス１０５からメッセージを受信し、ログファイル４１１にこのメッセージを書込む。プロセス１０５と後置型サーバ４０９との間の通信は、ＵＮＩＸオペレーティング・システムにより提供されるソケットによる。
【００５５】フォウラーらの前掲書及び米国特許出願第０８／００３７号明細書に詳細に説明されているように、メッセージ・ジェネレータ４０３は、一連のライブラリルーチンに関する実行可能コードに各プロセス１０５を動的にリンクさせることにより実現される。監視されているプロセス１０５は、ＵＮＩＸオペレーティング・システムにより提供される標準的なシステムに関するコードの代わりに、動的にリンクされたコードを実行する。
【００５６】動的リンクはプロセスが初期化された時に起こる。実行可能コードはプロセス初期化時に動的リンクされるので、このプロセスにより実行されているアプリケーションコードを変更又は再リンクする必要が無い。動的リンクは、米国，カリフォルニア州のマウンテンビューに所在するサン・マイクロシステムズ(Sun Microsystems)社から１９３３に発行された“共用ライブラリ(Shared Libraries)”と題する文献中に詳細に説明されている。
【００５７】所定のＵＮＩＸシステム呼出に関するライブラリルーチンにおけるコードはシステム呼出を呼出す。しかし、この他に、これは副作用も発生する。ビジュアルプロセスマネージャにおいて、この副作用はメッセージ４０６を包含する。図６はメッセージ４０６の詳細な内容を示す。このメッセージは次のような項目を含む。
【００５８】●ソースｐｉｄ(SPID)６０１：メッセージを生成したシステム呼出を行ったプロセス１０５のｐｉｄ，
●ソースマシーン(SMACH)６０３：プロセス１０５を実行するコンピュータの識別子，
●システム呼出名(Syscall name)６０５：プロセス１０５により行われたシステム呼出の名称，
●システム呼出引数(Syscall Args)６０７：プロセス１０５により行われたシステム呼出の引数，
●システム呼出結果(Syscall result)６０９：プロセス１０５により行われたシステム呼出の結果。
【００５９】例えば、オープンシステム呼出に関するメ

ッセージ４０６では、引数６０７は、オープンすべきファイルのパス名及びファイルがオープンされる際のモードを含む。結果６０９は、新たにオープンされたファイルのデバイス番号及びノード番号を含む。

【００６０】ビジュアルプロセスマネージャのユーザビューの説明において指摘されるように、ユーザは、セットアップコマンドにおいて、どのタイプのシステム呼出を監視すべきか指定する。この情報は各プロセス１０５用のマスク４０５に記憶される。動的にリンクされたライブラリにおけるルーチンは、メッセージを送信する前にプロセス１０５用のマスク４０５をチェックし、マスク４０５が、メッセージ４０６をこの種のシステム呼出について送信すべきことを示す場合にのみ、メッセージ４０６を送信する。

【００６１】好ましい実施例では、マスクはセットアップコマンドで設定され、システム呼出の種類について設定される。その後、マスクは監視されているプロセスの全てにより継承される。その他の実施例では、ユーザは、個々のプロセス及びシステム呼出についてマスクを設定するために、メニューを使用できる。好ましい実施例では、更に“簡潔(terse)”マスクを包含する。このマスクは、文字列をリターンするシステム呼出で指定できる。

【００６２】簡潔マスクがシステム呼出について設定されない場合、システム呼出は文字列をメッセージ４０６にリターンする。簡潔マスクが設定される場合、システム呼出はポインタ１１５を文字列にリターンする。プロセスのマスクが設定されると、セットアップコマンドを介すか又は対話型で、プロセスの子孫により継承される。

【００６３】ディスプレイジェネレータ４１９

ディスプレイジェネレータ４１９は、メッセージ解釈プロセス４１５とグラフ発生プロセス４１７の２つのプロセスとして実現される。好ましい実施例では、メッセージ解釈プロセス４１５は、コウトソフィオスらの前掲書に記載されたｌｅｆｔｙプログラムを実行し、エムデンらの前掲書に記載されたｄｏｔプログラムを実行する。

【００６４】メッセージ・インタープリータ４１５はログファイル４１１からメッセージ４０６を読出す。リアルタイムモードでは、メッセージ・インタープリータ４１５は、後置型サーバ４０９がメッセージ４０６をログファイル４１１に書込むのに応じて、メッセージ４０６を読出す。シングルステップモードでは、メッセージ・インタープリータ４１５は、監視されているシステム内のプロセス１０５が全て終了した後だけ、ログファイル４１１からメッセージ４０６を読出し始める。

【００６５】各メッセージについて、メッセージ・インタープリータ４１５はメッセージ４０６からシステム呼出のタイプを決定し、グラフデータストラクチャ４２１（グラフ２０１を示す）で必要な変更をメッセージにより行う。メッセージ４０６がもはや存在しない場合、または最後のリドローの時から所定の数のメッセージ４０６が処理された場合、ディスプレイ２０１はリドローされる。グラフ２０１がリドローされる場合に起こる事象は、メッセージ４０６により必要とされる変更の種類により左右される。

【００６６】変更がグラフ２０１の新たなレイアウトを必要としない場合、メッセージ・インタープリータ４１５はグラフデータストラクチャ４２１から情報を単に獲得し、この情報をウインドウシステム４２７に供給する。ウインドウシステム４２７はこの情報を使用し、表示装置１１２にグラフ２０１を実際に表示する。変更が新たなレイアウトを必要とする場合、メッセージ・インタープリータ４１５は新たなレイアウトに必要な情報をグラフデータストラクチャ４２１から獲得し、この情報を論理グラフ記述４２３として、グラフジェネレータ４１７に供給する。

【００６７】グラフジェネレータ４１７は、論理グラフ記述に基づき物理グラフ記述４２５をメッセージ・インタープリータ４１５にリターンする。その後、メッセージ・インタープリータ４１５は、新たな物理グラフ記述をグラフデータストラクチャ４２１に組込み、更新されたグラフデータストラクチャ４２１からの情報を前記のようにウインドウシステム４２７に供給する。

【００６８】図７は、グラフデータストラクチャ４２１の詳細な構成図である。グラフデータストラクチャ４２１はノードアレイ７０１とエッジアレイ７１３の２つの主要構成要素を有する。ノードアレイ７０１はグラフ２０１内の各ノードに関するノードエントリ７０３を包含する。エッジアレイ７１３はグラフ２０１内の各エッジに関するエッジエントリ７１５を包含する。各ノードエントリ７０３は、下記の４種類の情報を包含する。

【００６９】●論理ノード情報，メッセージ・インタープリータ４１５はこの情報を論理グラフデータ４２３としてグラフジェネレータ４１７に供給する，

●ノードディスプレイ情報７０７，グラフジェネレータ４１７はこの情報を、論理グラフデータ４２３に応答して物理グラフデータ４２５としてメッセージ・インタープリータ４１５に供給する，

●ビジュアルプロセスマネージャ情報７０９，及び

●エッジポインタリスト７１１。

【００７０】ビジュアルプロセスマネージャ情報７０９は、ディスプレイ２０１内のノード、及びノードエントリ７０３により示されるノードに付随するメニューに現れる情報である。エッジポインタリスト７１１は、ノードエントリ７０３により示されるノードで開始又は停止するエッジに関する各エッジエントリ７１５に対するポインタを包含する。

【００７１】エッジエントリ７１５は各エッジに関する同じ種類の情報を包含する。論理エッジ情報７１７は、

メッセージ・インターブリータ４１５がグラフジェネレータ４１７に供給したエッジに関する情報を包含する。エッジディスプレイ情報７１９は、グラフジェネレータ４１７によりメッセージ・インターブリータ４１５にリターンされる情報を包含する。ＶＰＭ情報７２１は、エッジに付随するメニューに現れる情報である。

【００７２】ポインタ７２３から(from ptr 723)は、エッジの送り手であるノードアレイ７０１内のノードエントリ７０３に対するポインタである。ポインタ７２５へ(toptr 725)は、エッジの受け手であるノードアレイ７０１内のノードエントリに対するポインタである。

【００７３】好ましい実施例で実現されるようなモニタ４１８の限界は、メッセージ・インターブリータ４１５が、プロセス１０５が停止したか否かメッセージ４０６から常に予測できるとは限らないことである。好ましい実施例では、この問題は、メッセージ・インターブリータ４１５に、所定のプロセス１０５から最後のメッセージ以来の時間を追跡させることにより処理される。

【００７４】或る一定の制限時間を超えた場合、メッセージ・インターブリータ４１５は、所定のプロセスが実行されるコンピュータについてｐｉｄサーバ４２９に照会し、プロセス１０５が依然として活動中であるか否か決定する。プロセス１０５が活動中でない場合、メッセージ・インターブリータ４１５は、プロセス１０５に関する情報をグラフデータストラクチャ４２１から削除する。そして、グラフ２０１の次のリドローの時に、プロセス１０５に関するノード及びプロセスにより単独でアクセスされるファイル又はパイプに関するノードが消失する。

【００７５】更に、メッセージ・インターブリータ４１５は、ウインドウシステム４２７により供給されるような、キーボード１１７及びマウス１１９からの入力を処理する。例えば、ユーザがマウスの右ボタンをクリックし、グローバルメニュー又はノードメニュウを要求すると、メッセージ・インターブリータ４１５は、要求されているメニューのタイプを決定し、このメニューに関する情報を、ウインドウを表示するウインドウシステム４２７に供給する。

【００７６】同様に、ユーザが、情報の表示を必要とするメニューエントリを選択する場合、ウインドウシステム４２７は、この選択が行われたことを示し、メッセージ・インターブリータ４１５は、グラフデータストラクチャ４２１からこの情報を取得し、この情報を表示するためにウインドウシステム４２７に供給する。ユーザが特定のプロセスに関するノードメニューについてキルエントリを選択する場合、メッセージ・インターブリータ４１５は、プロセス１０５を削除するシステムコマンドを実行する。

【００７７】ビジュアルプロセスマネージャ４０１の対話型バージョン

好ましい実施例では、端末におけるユーザはログファイル４１１内のメッセージがメッセージ・インターブリータ４１５により読出される速度を決定することだけができる。ユーザはビジュアルプロセスマネージャ４０１によりシステム内のプロセスの実行を制御できない。

【００７８】別の実施例では、ユーザはプロセスのシステムの実行を実際に制御するために、ビジュアルプロセスマネージャ４０１を使用できる。この実施例では、ログファイル４１１はソケット４２０により置換される。ソケット４２０は後置型サーバ４０９をメッセージ・インターブリータ４１５へ直接接続する。更に、双方向性ソケット４１６は、監視されているプロセス１０５用のメッセージ・ジェネレータ４０３と後置型サーバ４０９とを接続する。

【００７９】メッセージ４０６、マスク４０５及び動的リンクライブラリ用のコードでも変更が必要である。メッセージ４０６は追加フィールドのＡＣＫ（肯定応答）フィールド６１１を帯びる。このＡＣＫフィールド６１１は、メッセージ・インターブリータ４１５が後置型サーバ４０９から受信されたメッセージを肯定応答しなければならないか否か示す。マスク４０５は肯定応答マスクを包含する。肯定応答マスクは、システム毎の呼出ベースで、どのシステム呼出が肯定応答を必要とするか示す。

【００８０】最後に、動的リンクライブラリルーチンは、メッセージの送信前に肯定応答マスクを検査するコードを含むように変更される。マスクが、システム呼出が肯定応答を必要とすることを示す場合、システム呼出を置換するライブラリルーチンはメッセージ内の肯定応答フィールド６１１をセットし、そして、後置型サーバ４０９からの肯定応答メッセージを待機する。メッセージが到着した後にだけ、ライブラリルーチンはその実行を完了する。

【００８１】対話型モードの動作について説明する。プロセス１０５を対話型モードにするために、ユーザは前記のようなマウスを使用しプロセスのメニューを取得する。メニュー５０１は対話型モードを指定する追加エントリを有する。ユーザがこのエントリを選択すると、別のメニューが現れる。この別のメニューにより、ユーザは、どのシステム呼出をこのプロセスについて肯定応答すべきか指定できる。メッセージ・インターブリータ４１５はメニューから入力を獲得し、どのシステム呼出は現に肯定応答されているか示すノードのためのフィールドを設定する。

【００８２】メッセージ・インターブリータ４１５はまた、ソケット４２０を介してメッセージを後置型サーバ４０９に送信する。このメッセージはシステム呼出が肯定応答されるべきプロセス１０５及び肯定応答されるべきシステム呼出を指定する。その後、後置型サーバ４０９はソケット４１６を介して、メッセージを指定プロセ

ス１０５に送信し、メッセージ・ジェネレータ４０３内のコードは、メッセージにより必要とされるような肯定応答マスクを設定することによりコードに応答する。

【００８３】肯定応答マスクが設定された時点から、システム４０１は次のように動作する。プロセス１０５がシステム呼出を実行すると、このシステム呼出用のライブラリルーチンはシステム呼出用の肯定応答マスク４０５をチェックする。マスクが設定されていれば、メッセージ４０６はその肯定応答フィールド６１１設定を有し、ライブラリルーチンはシステム呼出の実行及びメッセー４０６の送信後、休止する。

【００８４】メッセージがソケット４１６、後置型サーバ４０９及び及びメッセージ・インターブリータ４１５内のソケット４２０から到着すると、メッセージ・インターブリータ４１５は、メッセージ４０６により必要とされるようなグラフ２０１の変更及びディスプレイ内のプロセス用のプロセスボックス２０５の外観の変更により、設定肯定応答フィールド６１１に応答し、プロセスボックス２０５により示されるプロセス１０５は肯定応答を待機中であることを示す。

【００８５】ユーザはポインタ１１５をプロセスボックス２０５に移動させ、マウスの左ボタンをクリックすることにより、肯定応答を与える。ポインタ１１５の位置及びマウス左ボタンのクリックに応答して、メッセージ・インターブリータ４１５はソケット４２０，後置型サーバ４０９及びソケット４１６を介して肯定応答メッセージをこのプロセス用のメッセージ・ジェネレータ４０３に送信する。このメッセージに応答して、ライブラリルーチンはその実行を終了する。これが行われている最中に、メッセージ・インターブリータ４１５は更にグラフ２０１の外観を変更し、プロセスはもはや肯定応答を待機していないことを示す。

【００８６】前記の説明から明らかなように、システム４０１のユーザは、プロセス４０１の実行を任意の所望の粒度にまで制御することができる。その他の実施例も、肯定応答マスクのグローバル設定を可能にするグローバルメニュー３０１内にエントリを有することができる。

【００８７】結論

以上、本発明のビジュアルプロセスマネージャの実現に関する好ましい実施例について説明した。しかし、前記のように、ビジュアルプロセスマネージャの実現に使用される技法は、プロセスのシステム解析に限定される手段ではなく、システムにより使用されるライブラリが別のライブラリにより動的に置換される全てのシステムで使用することもできる。

【００８８】当業者には明らかなように、前記のビジュアルプロセスマネージャを実現する方法は他にも多数存在する。例えば、前記に説明した動的リンク技法及びグラフ作図技法は本発明の実現に特に有用であるが、モニタ４１８内のライブラリルーチンは、ディスプレイジェネレータ４１９を実現するのに使用できるｌｅｆｔｙ及びｄｏｔプログラム以外のプロセス及びグラフ作図プログラムにより実行されるコードに静的にリンクさせることもできる。

【００８９】監視すべきオペレーティング・システム呼出の選択は、オペレーティング・システム毎に変化し、また、ビジュアルプロセスマネージャの企図される用途に応じても変化する。同様に、利用可能なマスキングの種類は実現される対象毎に異なる。

【００９０】同一の系統では、好ましい実施例におけるグラフ２０１で使用される形状，ラベル及び色彩が特に有用であるが、他の実施例では、他の形状及び他の色彩も使用でき、更に、どのような情報をノードに直接示すか、及び何がメニューによりアクセス可能かなどに関して異なる決定を行うこともできる。また、他の実施例では、ユーザに、ディスプレイ２０１の制御又はプロセスのシステムの操作の制御を多少は提供することもでき、システムの実行を開始させるために異なる方法を使用することもできる。

【００９１】

【発明の効果】以上説明したように、本発明の方法によれば、コンピュータで実行するプロセスを構成するシステムの精密な解析と制御が可能になる。また、共同プロセスを構成するデバッキングシステムにおいて資源を有効に活用することができる。また、本発明の方法はシステム解析の他に、ルーチンのライブラリが、同じ機能を果たすが副作用としてメッセージも送信する動的リンクライブラリにより置換され得るような全ての状況において使用することができる。本発明の方法の特別な効果は、システムのアプリケーションレベルコードの変更又はリコンパイルすること無しに、システムを解析することができることである。

【図面の簡単な説明】

【図１】本発明を実現することができる分散型コンピュータシステムの一例の模式的構成図である。

【図２】本発明により生成された表示の一例の模式図である。

【図３】本発明により生成された表示の別の例の模式図である。

【図４】ビジュアルプロセスマネージャの好ましい実施例の構成を示す概要ブロック図である。

【図５】本発明により生成された表示の更に別の例の模式図である。

【図６】本発明で使用されるメッセージのブロック図である。

【図７】本発明で使用されるグラフデータストラクチャのブロック図である。

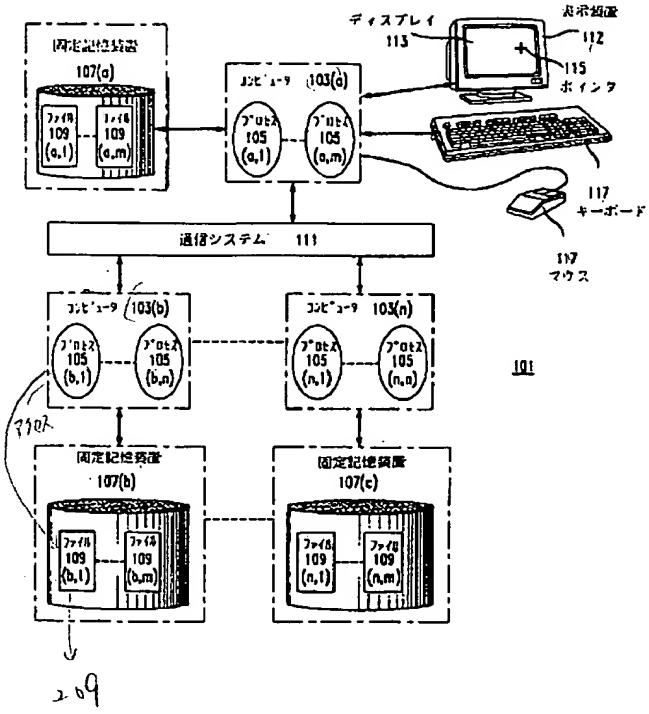
【符号の説明】

１０３　プロセッサ

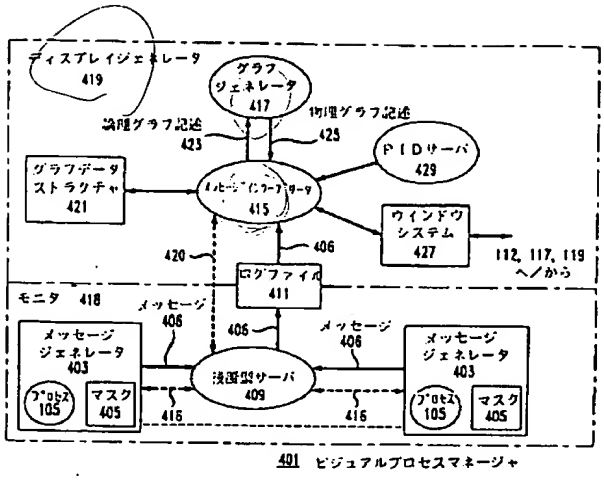
19　　20

１０５　プロセス
１０７　固定記憶装置
１０９　ファイル
１１１　通信システム
１１２　表示装置
＊１１３　ディスプレイ
１１５　ポインタ
１１７　キーボード
１１９　マウス

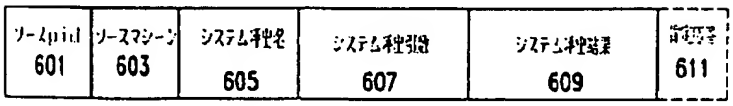
【図１】



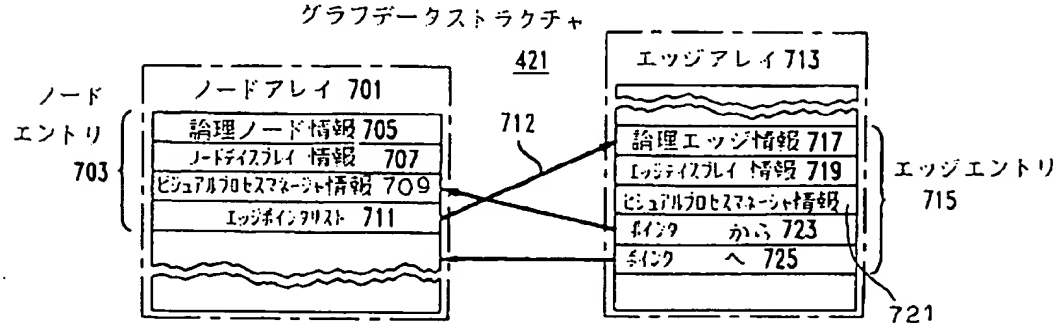
【図４】



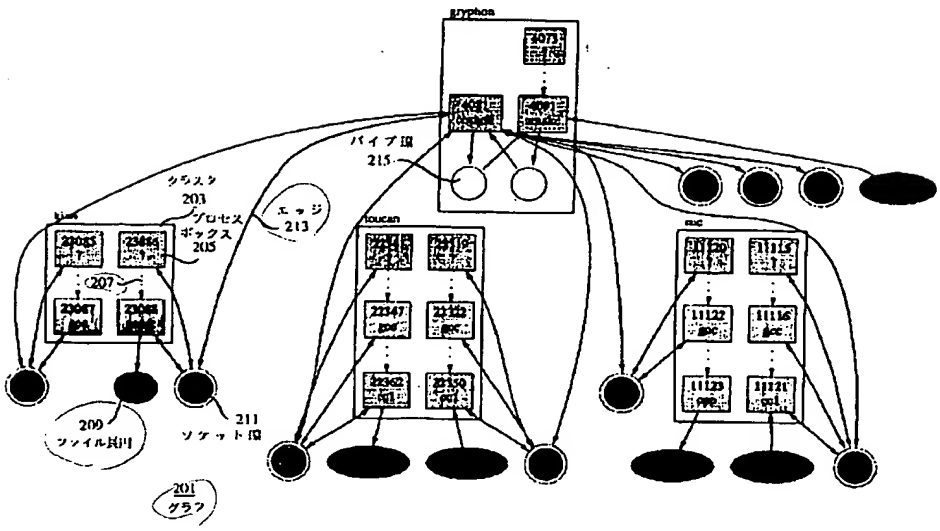
【図６】



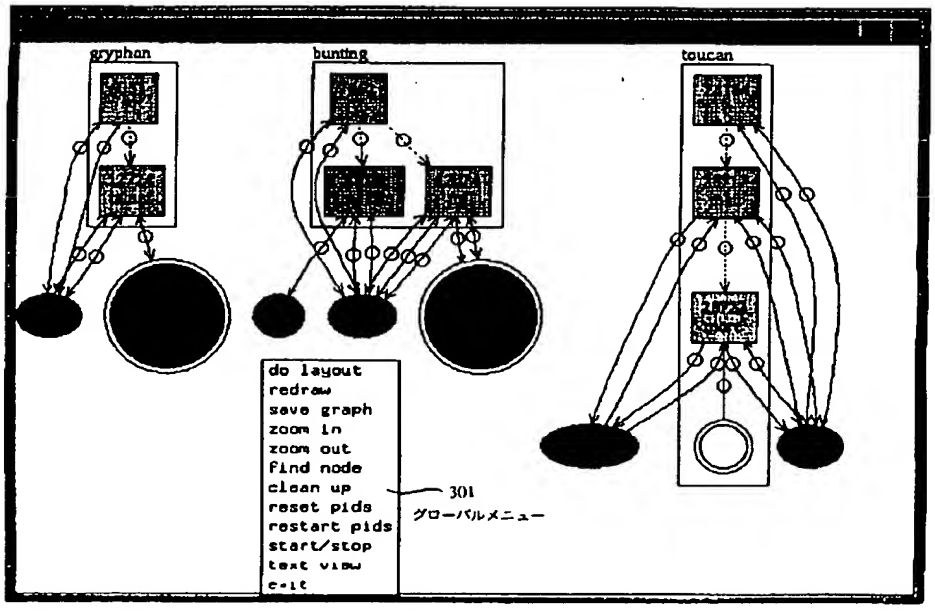
【図７】



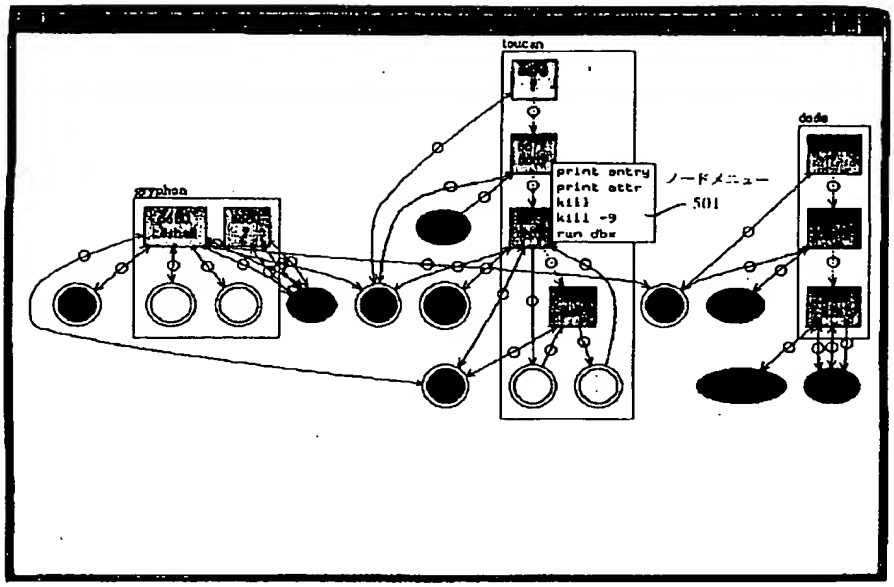
【図２】


gryphon
toucan
パイプ図
215
クラスタ
203
プロセス
ボックス
205
エッジ
213
207
209
ファイル図
211
ソケット図
201
グラフ

【図３】


gryphon
bunting
toucan
do layout
redraw
save graph
zoom in
zoom out
find node
clean up
reset pids
restart pids
start/stop
text view
exit
301
グローバルメニュー

【図５】



フロントページの続き

(72)発明者　デビッド　ジェラード　コーン
アメリカ合衆国、10003　ニューヨーク、ニューヨーク、エー　107、マーサー　ストリート　303

(72)発明者　エレフテリオス　コウツオフィオス
アメリカ合衆国、07928　ニュージャージー、チャタム、アプト．イー４、リバーロード　420

(72)発明者　ステファン　シー．ノース
アメリカ合衆国、07830　ニュージャージー、カリフォン、フィルハウアー　アベニュー　127